40th Anniversary Edition

DATABASE PROCESSING

FUNDAMENTALS, DESIGN, AND IMPLEMENTATION

David M. Kroenke | David J. Auer | Scott L. Vandenberg | Robert C. Yoder

FIFTEENTH EDITION

OTHER MIS TITLES OF INTEREST

Introductory MIS

Experiencing MIS, 8/e
Kroenke & Boyle ©2019

Using MIS, 10/e
Kroenke & Boyle ©2018

Management Information Systems, 15/e
Laudon & Laudon ©2018

Essentials of MIS, 13/e
Laudon & Laudon ©2019

Processes, Systems, and Information: An Introduction to MIS, 3/e
McKinney & Kroenke ©2019

Information Systems Today, 8/e
Valacich & Schneider ©2018

Introduction to Information Systems, 3/e
Wallace ©2018

Database

Hands-on Database, 2/e
Conger ©2014

Modern Database Management, 13/e
Hoffer, Ramesh & Topi ©2019

Database Concepts, 8/e
Kroenke, Auer, Vandenberg & Yoder ©2018

Database Processing, 15/e
Kroenke, Auer, Vandenberg & Yoder ©2019

Systems Analysis and Design

Modern Systems Analysis and Design, 8/e
Hoffer, George & Valacich ©2017

Systems Analysis and Design, 10/e
Kendall & Kendall ©2019

Decision Support Systems

Business Intelligence, Analytics, and Data Science, 4/e
Sharda, Delen & Turban ©2018

Business Intelligence and Analytics: Systems for Decision Support, 10/e
Sharda, Delen & Turban ©2014

Data Communications & Networking

Applied Networking Labs, 2/e
Boyle ©2014

Digital Business Networks
Dooley ©2014

Business Data Networks and Security, 11/e
Panko & Panko ©2019

Electronic Commerce

E-commerce 2018: Business. Technology. Society, 14/e
Laudon & Traver ©2019

Enterprise Resource Planning

Enterprise Systems for Management, 2/e
Motiwalla & Thompson ©2012

Project Management

Project Management: Process, Technology and Practice
Vaidyanathan ©2013

40th Anniversary Edition

FIFTEENTH EDITION

DATABASE PROCESSING
FUNDAMENTALS, DESIGN, AND IMPLEMENTATION

David M. Kroenke

David J. Auer
Western Washington University

Scott L. Vandenberg
Siena College

Robert C. Yoder
Siena College

Pearson

330 Hudson Street, NY NY 10013

Vice President, IT & Careers: Andrew Gilfillan
Senior Portfolio Manager: Samantha Lewis
Managing Producer: Laura Burgess
Associate Content Producer: Stephany Harrington
Portfolio Management Assistant: Madeline Houpt
Director of Product Marketing: Brad Parkins
Product Marketing Manager: Heather Taylor
Product Marketing Assistant: Jesika Bethea
Field Marketing Manager: Molly Schmidt
Field Marketing Assistant: Kelli Fisher
Cover Image: Cover art "waterfall" by Donna Auer

Vice President, Product Model Management: Jason Fournier
Senior Product Model Manager: Eric Hakanson
Lead, Production and Digital Studio: Heather Darby
Digital Studio Course Producer: Jaimie Noy
Program Monitor: SPi Global
Full-Service Project Management and Composition: Cenveo® Publisher Services
Printer/Binder: LSC Communications
Cover Printer: Phoenix
Text Font: 10/12 Mentor Pro

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on the appropriate page within text.

Library of Congress Cataloging-in-Publication Data

Names: Kroenke, David M., 1948- author. | Auer, David J., author. |
Vandenberg, Scott L., author. | Yoder, Robert C., author.
Title: Database processing : fundamentals, design, and implementation /David
M. Kroenke, David J. Auer, Western Washington University, Scott L.
Vandenberg, Siena College, Robert C. Yoder, Siena College.
Description: 15th edition, 40th anniversary edition. | Boston : Pearson,
[2018] | Includes bibliographical references and index.
Identifiers: LCCN 2017041164| ISBN 9780134802749 | ISBN 0134802748
Subjects: LCSH: Database management.
Classification: LCC QA76.9.D3 K7365 2018 | DDC 005.74–dc23 LC record available
at https://lccn.loc.gov/2017041164

Pearson

ISBN 10: 0-13-480274-8
ISBN 13: 978-0-13-480274-9

Brief Contents

This page intentionally left blank

Contents

Foreword to the 40th Anniversary Edition

David Kroenke

The publisher has asked me to write a short history of this text for this, the 40th anniversary edition. The details of each edition and how they changed are instructive, but this text and the discipline of database processing grew up together, and the story of how that happened might be more helpful to students who will work in disciplines, such as Big Data, that are emerging today.

We Didn't Know What We Were Doing

Database processing technology originated in the period 1970 to 1975, though not necessarily by that name. At the time, the U.S. government used the term *data bank*. Others used *data base* as well as *database*. I liked the latter and used it when I began work on this text in 1975.

In 1971, I was an officer in the U.S. Air Force, assigned to a Pentagon team that was building and using a simulation of World War III. It was the height of the Cold War, and the Department of Defense wanted a means to assess the efficacy of current and proposed weapons systems.

By a stroke of good luck, I was assigned to work on the data manager portion of that simulation (the term *Database Management System [DBMS]* was not yet in use). The logical data model of that data manager was similar to that of the set-based system that Bachmann had developed at General Electric (then a mainframe manufacturer) and that later became the CODASYL DBTG standard.[1]

Our simulation was slow and long-running; a typical run would take 10 to 12 hours. We were constrained more by input and output of data than by CPU time, and I developed low-level, re-entrant, assembly language routines for getting and putting data to and from main memory on parallel channels.

In addition to our project and Bachmann's, IBM was developing a manufacturing-oriented data manager in concert with North American Aviation. That project eventually became IBM's product IMS.[2] Another government project of that era resulted in the data manager named Total.

In retrospect, I'd say the one thing we had in common was that none of us knew what we were doing. We didn't have any data models, best practices, or design principles. We didn't even know how to program. This was long before GoTo-less programming, which led to structured programming, and eventually to object-oriented programming. We did know that life was easier if we developed some sort of a logic chart before we began, but that was about it. We'd pick up our coding pads (everything was done via punched cards) and start to work.

There were no debugging tools. When a job would fail, we'd receive a hexadecimal printout of the CPU registers and the contents of main memory (the printout would be 12 to 18 inches thick). There were no hexadecimal calculators, so we'd manually add and subtract

[1] CODASYL, the Committee on Data Systems Languages, was the committee, chaired by Grace Hopper (see *https://en.wikipedia.org/wiki/Grace_Hopper*), that developed the COBOL language standard. DBTG, the database task group, was a subcommittee tasked with developing a data modeling standard. The DBTG model was popular for a short while, but was replaced by the relational model by the 1980s.

[2] IBM IMS is still a functional DBMS product—see *www-01.ibm.com/software/data/ims/index.html.*

hexadecimal numbers to navigate our way around the printout, sticking rulers in the listing as place markers. Stiff, wooden rulers were the best.

Again, though, we were just trying to solve a problem. We didn't have any idea that the technology we were developing would become an important part of the emerging world. Imagine Amazon or your college without database processing. But all of that was in the future. We were just trying to get the "darn thing" to run and somehow solve the particular problem that we'd been assigned.

For example, an important function of those early systems was to manage relationships. In our simulation, we had bombers and tankers and opposing radar sites and opposing air-to-air missiles. We needed to keep track of which of those was assigned or related to which. We just wrote programs to do that. A decade or two later someone discovered in surprise, "Hey, there's as much information in the relationships as there is in the data."

We made stuff up as we went along. The first edition of this text included no definition of *database*. When a reviewer pointed that out, I made one up for the second edition. "A self-describing collection of integrated records." Completely fabricated, but it's worked now for 35 years, so it must have been serviceable.

Situations like that were common in those early projects. We made stuff up that would help us solve our problem. Progress was slow, mistakes were frequent, failures were common. Millions of dollars and labor hours were wasted. But gradually, over time, database technology emerged.

Origin of This Text

In 1973 I completed my military commitment and following John Denver's song "Rocky Mountain High" moved my family from Washington, D.C., to Colorado State University. The business school hired me as an instructor while I attended graduate school in statistics and engineering across the street. To my delight, I was assigned to teach a course entitled File Management, the predecessor of today's Database Processing course (see Figure FM-1).

As with any young instructor, I wanted to teach what I knew and that was the rudiments of database processing. So, I began to formulate a database course and by the spring of 1975, was looking for a textbook. I asked the book reps if they had such a book and none did. The sales rep for SRA, however, asked, "No, but we're looking for one. Why don't you write it?" My department chair, Bob Rademacher, encouraged me to do so, and on June 29, 1975, I signed the contract.

FIGURE FM-1

David Kroenke Loses Control of Students Excited by Database Technology

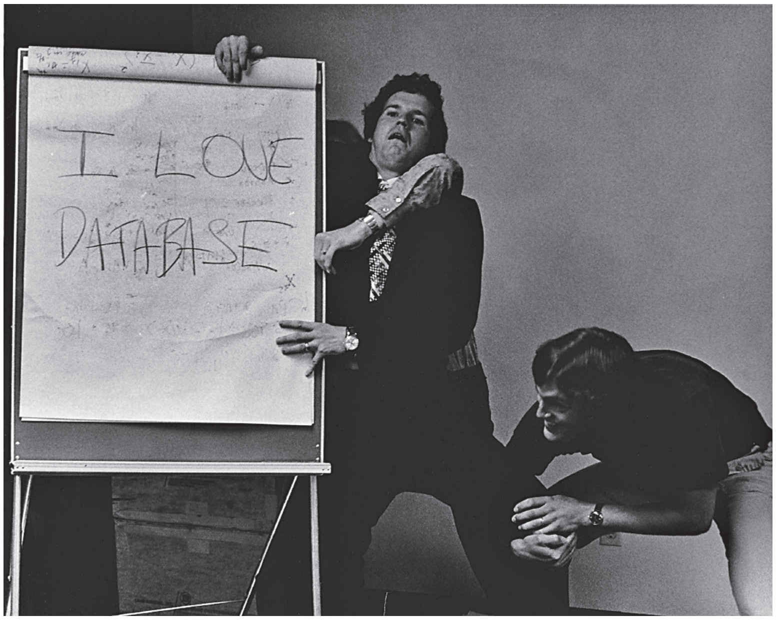
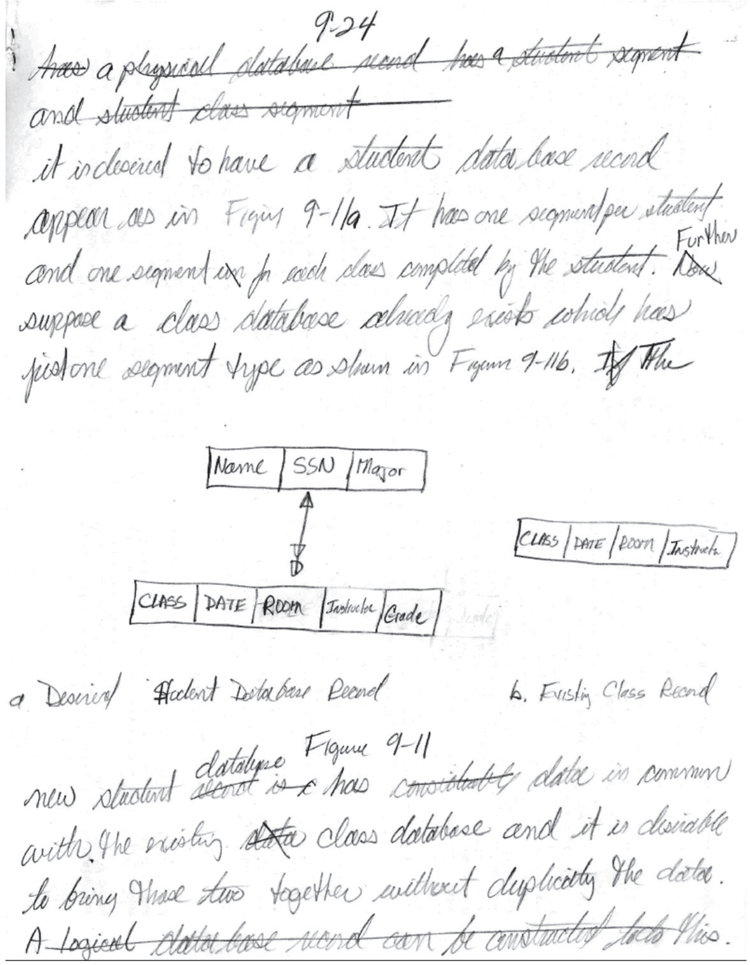


FIGURE FM-2
How Textbooks Were Written


9-24
Has a physical database record has a student segment
and student class segment
it is desired to have a student data base record
appear as in Figure 9-11a. It has one segment per student
and one segment for each class completed by the student. Further
suppose a class database already exists which has
just one segment type as shown in Figure 9-11b. The

Name | SSN | Major
CLASS | DATE | ROOM | Instructor | Grade
CLASS | DATE | ROOM | Instructor

a. Desired Student Database Record
b. Existing Class Record
Figure 9-11

new student database has data in common
with the existing class database and it is desirable
to bring these two together without duplicating the data.
A logical database record can be constructed to do this.

The draft and all the diagrams were written in number 2 pencil on the back of old coding sheets, as shown in Figure FM-02. The text would go to a typist, who'd do the best she could to decipher my writing. I'd proof the typing and she'd produce another typed manuscript (long before word processing–pages had to be retyped to remove errors). Those pages would then go to a copy editor and I would redo them again, back to the typist for a round or two. Eventually, the final typed manuscript would go to a compositor who would produce long gray sheets (called galleys) of text to be proofed. After that, the text would be glued (I'm not kidding) to make up pages, integrating the art which had been following a similar pathway, and then those pages would be photographed and sent to a printer.

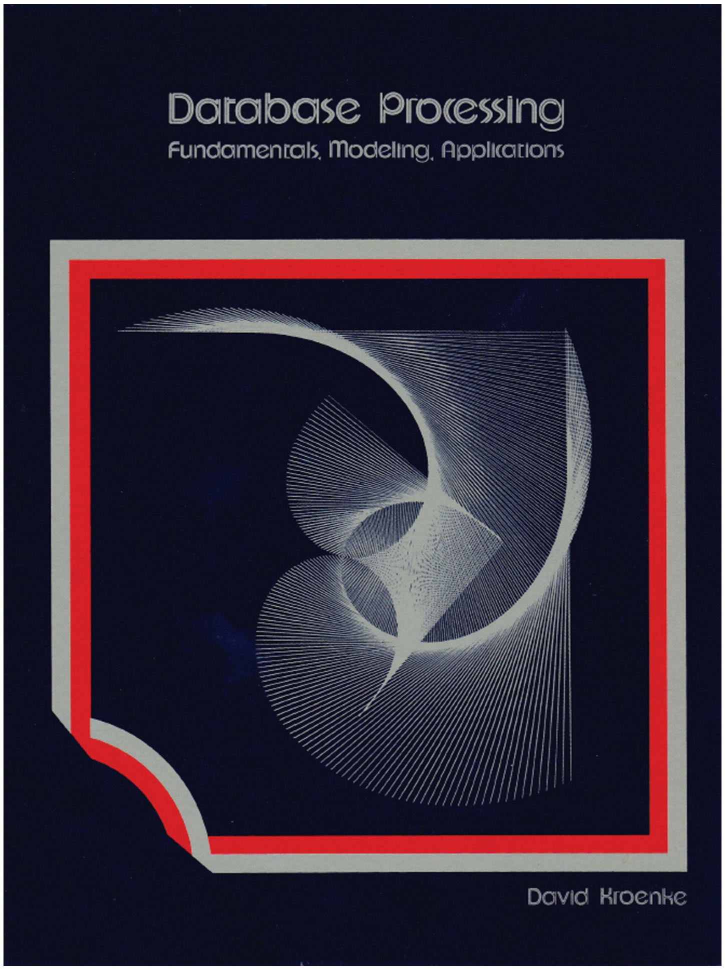
The final draft of the first edition was completed in January 1976, and the text was published in January 1977. We were proud that it only took a year.

Contents of the First Edition

Database Processing was the first such textbook aimed at the information systems market. C. J. Date had produced *Database Systems* prior to this text, but his book was aimed at computer science students.[3] No one knew what should be in an information systems database book. I made it up, we sent drafts to reviewers, and they approved it. (They didn't know either.)

[3] C. J. Date's book *An Introduction to Database Systems* is currently in its eighth edition.

FIGURE FM-3
Cover of the First Edition of Database Processing



The first edition (see Figure FM-3) had chapters on file management and data structures. It also had chapters on hierarchical, network, and relational data models. By the way, E. F. Codd, the creator of the relational data model, was relatively unknown at that point and he was happy to review the relational chapter. The text also featured a description of the features and functions of five DBMS products: ADABAS, System 2000, Total, IDMS, and IMS. (To my knowledge, only IMS is still in use today.) It wrapped up with a chapter on database administration.

When writing that last chapter, I thought it would be a good idea to talk with an auditor to learn what auditors looked for when auditing database systems. Accordingly, I drove to Denver and met with one of the top auditors at one of the then-Big-Eight firms. I didn't learn much, just some high-level hyperbole about using commonly accepted auditing standards. The next day, the phone rang in my office and an executive in New York City invited me out to that firm for a job interview for a position to develop and teach database auditing standards to their staff. None of us knew what we were doing!

I had no idea of how incredibly fortunate I was. To stumble into a discipline that would become one of the most important in the information systems field, to have experience and knowledge to put into a text, to have a supportive department, and, finally, to have what was at that time a superb publisher with an outstanding sales and marketing team (see Figure FM-4). Because it was all I had known, I thought it was normal. Ah, youth.

Lessons Learned

At age 71, I'm not quite consigned to watching the daytime weather channel but have reached the stage when people start listing lessons learned. I'll try to keep it brief. Here are my five lessons learned, developed both as a database technology bystander and participant:

FIGURE FM-4
Hot Marketing Handout 1977—Note Text Price

SRA SCIENCE RESEARCH ASSOCIATES, INC. · College Division – 1540 Page Mill Road, Palo Alto, California 94304 (415) 493-4700

DATABASE PROCESSING:

Fundamentals, Modeling, Applications

David M. Kroenke, Colorado State University

DESCRIPTION:
13-4100 Text, Hard Cover, 416 pp
13-4101 Instructor's Guide

AVAILABLE:
$16.95 List (Tentative)

SUGGESTED COURSES:
Database Processing, Data Management, Information Storage and Retrieval, Database Systems, Database Management; in Departments of Business, Information Systems, Management Science, Accounting.

OVERVIEW:
An introduction to database management specifically *designed for classroom use.* The extensive apparatus includes projects, chapter summaries, review and discussion questions, numerous illustrations, review tables, short case materials, and an optional student-oriented database package. The scope of the text is unusually broad. It addresses fundamentals (including file design and data structures), modeling (DL/1, CODASYL DBTG, and Relational), and applications (database system, database analysis and design, and database administration). Students will find the text very readable and will appreciate the student-related examples used. In addition, special terms are explained throughout; over 250 questions and projects are presented.

Topic coverage is logically organized. It progresses from the basics of data storage (which provides a common ground for understanding), through database processing (which contains material and concepts that are new to most readers), to database implementation. The final chapter discusses specific application cases drawing together all the textual material.

The author's varied experience in the field contributes greatly to the effectiveness of the text. During his years in industry, the author designed, developed, and implemented a database system. He serves as a consultant and professional lecturer on database methodology concepts and problems and for three years, has taught a database course.

Don't Confuse Luck with Exceptional Ability

According to an independent study at the time, the second edition of this text had 91 percent of the market. It was the first, and it had a great publisher with a superb sales force. That success was due, truly, to lucky timing and good fortune. At that point I should have doubled down and made sure that the 91 percent were satisfied while sending Thanksgiving turkeys to the 9 percent not in the fold.

Instead, what did I do? Ignored the book and joined Microrim to help develop the R:Base products.[4] Five years later when I returned to the book, numerous competitors had emerged and the book lost half of its market. I'd thought I could jump back in and regain that early success, but the market had a hard lesson for me.

I mention this because I've seen it elsewhere as well. Microsoft was built and managed by superior business professionals like Bill Gates, Steve Ballmer, Jon Shirley, Steve Okey, and, in the database domain, David Kaplan. Between 1985 and 2000, hundreds of employees joined the firm and were issued stock options. They were largely competent professionals, but no different from the same level of professionals one would have found at 3M, Procter and Gamble, Boeing, etc. The difference was that during that interval, their Microsoft stock split seven times.

Many of those competent professionals understood that they had been very, very lucky to get on the Microsoft bus when they did. They took their stock proceeds and re-invested in index funds or something else safe and have been enjoying life on the golf course with their

[4] For more information on R:Base, see the Wikipedia article R:Base (*https://en.wikipedia.org/wiki/R:Base*). Now called RBASE, this is still a functional DBMS product–see *http://www.rbase.com/*.

families ever since. However, some of the just-competent professionals confused their good luck with exceptional personal ability and founded their own companies or started venture capital firms. Most lost their money. They were good, but they weren't of the same caliber as Gates et al.

Joseph Conrad said it, "It is the mark of an inexperienced man not to believe in luck."

Marketing Trumps Technology

If you have a chance to invest in an average technology with superb marketing or superb technology with average marketing, take the former. Marketing is far more important than technology.

IBM's IMS uses a hierarchical data model. Representing many-to-many relationships with hierarchies is a pain. With IMS the database developer is forced to write all sorts of design and coding machinations that should have been done by the DBMS or avoided by using a different DBMS. In the early days, I watched an IBM technical sales representative present those machinations as a skill that every good database developer must already have. "Surely you know how to do the XYZ?" (I don't remember the name they'd given that dance). Because none of us in the audience knew any better, we all assumed that we were deficient if we didn't know how to do the XYZ. The deficiency was in the product, not us, but we were duped by good marketing.

Developed by Wayne Ratcliff, Ashton-Tate's dBase[5] was the most successful relational microcomputer DBMS product until Microsoft entered the picture with Microsoft Access. In fact, dBase was neither a DBMS nor was it relational—it was a file management system. However, Ashton-Tate's marketing convinced Osborne to place a free copy of dBase on every one of its Osborne II computers. The Osborne II was the micro or personal computer (PC) of choice for a new cadre of application developers, and they wrote millions of lines of dBase code. They used what they had and thought it was fine. When better products came along, there was no way that any small developer was going to rewrite existing code or learn new products. The new graphical user interface in Microsoft Windows, the Microsoft Office package, and cheap Microsoft Access pricing was the only force strong enough to push Ashton-Tate off its leading position.

Salsa, a product that I helped Wall Data develop, implemented the semantic object model (which we discuss in Appendix F), and was selected as the runner-up to Netscape Navigator for product of the year in 1996 (the other runner-up was Internet Explorer). Salsa failed—not because of the technology, but because of the marketing. We tried to sell it as an end-user product and it was a developer product. It was as if we'd invented Gore-Tex and we were out trying to sell it to people standing in the rain. We should have sold it to the clothing manufacturers. Marketing 101. I still have nightmares about the superior technology that washed down that drain.

Christensen's Model Informs

I don't know anyone who made substantial money in mainframes that did well in the microcomputer industry. Accustomed to the features and power of mainframes, we viewed microcomputers as toys. We termed the TRS-80 micro the Trash-80. I bought an early Apple and it crashed on me and I thought to myself, "This will never amount to anything."

This phenomenon is addressed by Clayton Christensen is his disruptive innovation model.[6] His thesis is that when a disruptive technology comes along, companies that have success in the disrupted technology are unable to capitalize on the opportunities of the new technology. Kodak could not adapt to digital photography; Swiss watch makers could not adapt to digital watches. Textbook publishers could not adapt to book rentals and used books sales by Amazon. Microsoft lost its way when it achieved its goal of "A computer on every desk and in every home." It struggled to adapt to the Internet.

---

[5] dBbase is still a functional DBMS product—see *http://www.dbase.com*

[6] Christensen, Clayton M. *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail* (Reprint edition). (Brighton, MA: Harvard Business Review Press, 2106)

Don't look for the market leaders in big data or robotics to come from existing, large vendors. They will come from smaller companies that can position themselves to thrive in the new environment. If you haven't learned Christensen's model, you should.

Good-enough Fashion Will Do

For the most part, the relational model supplanted all the other data models. Because of Codd's insights on the use of functional dependencies for relational design, it provided sound design principles. Also, fixed-length records (as, in practice, they were at first) fit nicely with existing file management capabilities. It worked. Thousands of papers were written on the topic.

However, today's technology readily supports the storage and searching of multiple fields in un-normalized documents. In 1980, technology constraints required designers to take a document like a sales order and break it up into its pieces: Invoice, Salesperson, Customer, Line-item, Product. They would then store those pieces and then put them all back together with SQL (Structured Query Language) when someone wanted the original sales order. That makes no sense today. It's like driving your car into a parking garage and having staff pull off your front tires and put them in the pile of front tires, your steering wheel into a pile of steering wheels, etc. Then, when you come back, put it all back together. Even though it's unnecessary, it's happening right now in zillions of data centers.

So why is the relational model still in use? Because it's good enough and still in fashion.

Fashion is important. Consider normalization theory. Codd's first paper addressed normalization through third normal form. However, in later papers, he and others showed that this wasn't enough. Relations in third normal form still had anomalies, which led to fourth and fifth and then Boyce-Codd (BCNF) Normal Forms. Despite this, one still hears people talk about third normal form as the be-all, end-all of relational design. Third normal form is good enough and still in fashion. It was as if progress stopped at third normal form (not in this text, though, where all of these forms are taught).

I suspect that someday soon, the whole relational mess will no longer be good enough and we'll move to XML or JSON or some other form of document storage (as we discuss in Chapter 12, Appendix I, and Appendix L). I've been saying that for 10 years, though, and it hasn't happened yet.

Another example of good enough and fashion is the entity-relationship (ER) model. The ER model is nothing more than a thin cover over the relational model. Entities are essentially logical relations, and relationships are a slim version of foreign keys. ER operates at too low a level of abstraction. Other models like the semantic object model and other object-oriented models are better. None succeeded. The ER model was in fashion and good enough.

When It's Over, It's Over

By the turn of the century, I'd been writing and revising this text for 25 years. Although I was exceedingly grateful to the thousands of professors and students who had used this book over those years, I also knew I was done. Partly, I said to myself, because it had settled down, the early crazy days were long gone, and partly because 25 years is a long time to work on a textbook.

To my great good fortune, I found David Auer, who agreed to take over the revisions of this text. I am most grateful to David for his hard work and for his fidelity to the underlying goals and philosophy of this text. The rest of this story is his.

David Auer

I was introduced to David Kroenke while working on the Instructor's Manual for the ninth edition of *Database Processing*. Because we were both living and teaching in western Washington State, we could get together for meals and discussions. This led to my working on the companion textbook, *Database Concepts*, being a technical reader for the 10th edition of *Database Processing*, and then being asked to become a coauthor for the 11th edition of *Database Processing*.

I am very fortunate to be able to work with David on these projects. If you have read his portion of this foreword, you will have gotten a brief glimpse into the mind of a very creative and articulate person. He was also in the right place at the right time to be part of the creation of the computer-driven world that we live in and work in today. He has made many important contributions over his career, and the book you are reading is certainly one of them—the first textbook on database systems for management information systems classes and still one of the leading textbooks in the field!

David constantly revised and expanded *Database Processing* as new topics became relevant. My main contributions to the 11th and following editions were making MySQL a DBMS discussed on the same level as Microsoft SQL Server and Oracle Database; formalizing the treatment of Web database applications; and introducing new current topics such as non-relational databases, Big Data, and cloud computing. I have also revised and updated our treatment of Structured Query Language, while maintaining the practical and "immediately usable on your own computer" presentation that has always been a hallmark of this book.

The main challenge now is to keep the book current with the changing technology and techniques in our app-driven, Internet and cloud computing world of today, where databases are used ubiquitously to support applications such as Facebook, Twitter, and Instagram. To this end, we have brought two new coauthors on board for this edition: Scott Vandenberg and Robert Yoder, who are researching and teaching these topics.

Although this 15th edition of *Database Processing* marks the 40th anniversary of the book, we look forward to providing you with many more years of current, accurate, and usable knowledge about the world of databases and how they are used.

Preface

The 15th edition of *Database Processing: Fundamentals, Design, and Implementation* refines the organization and content of this classic textbook to reflect a new teaching and professional workplace environment. Students and other readers of this book will benefit from new content and features in this edition.

New to This Edition

Content and features new to the 15th edition of *Database Processing: Fundamentals, Design, and Implementation* include the following:

- The reorganization of SQL topics in Chapter 2 has been kept and a section on SQL queries on recursive relationships has been added.
- The material on *Big Data* and the evolving *NoSQL movement* is summarized in Chapter 12 and then expanded upon in restructured Appendix J, "Business Intelligence Systems," Appendix K, "Big Data," and a new Appendix L, "JSON and Document Databases. " This is an important topic that is constantly developing and changing, and the new appendix structure provides room for an extended discussion of the topic. Material on virtualization and cloud computing is expanded and updated in Chapter 12. The chapter has also been revised to tie together the various topics of the chapter and give a more complete, contextualized treatment of Big Data and its various facets and relationships to the other topics.
- Online chapters on Microsoft SQL Server 2017 (Chapter 10A), Oracle Database (Chapter 10B), and MySQL 5.7 (Chapter 10C) now have a section on importing data from Microsoft Excel 2016 worksheets.
- The book has been updated to reflect the use of Microsoft SQL Server 2017, the current version of Microsoft SQL Server. Microsoft has made SQL Server Developer Edition (a one-user version of SQL Server Enterprise Edition) available for download at no cost, and therefore we use this Developer Edition instead of the Express Edition as the basis for our work with SQL Server in the book. Although most of the topics covered are backward compatible with Microsoft SQL Server 2016 and earlier versions, all material in the book now uses SQL Server 2017 in conjunction with Microsoft Office 2016 exclusively.
- Oracle's Oracle Database is now updated to Oracle Database 12*c* Release 2, and Oracle Database Express Edition 11*g* Release 2 (Oracle Database XE) is introduced as the preferred Oracle Database product for use on personal computers. In addition, a complete set of instructions for downloading, installing, and configuring Oracle Database 12*c* Release 2 (Enterprise or Personal Edition) has been added. The current version of the Oracle SQL Developer GUI tool provides a common interface to both versions of Oracle Database, and we provide detailed examples of how to use it.
- Online Chapter 10C, "Managing Databases with MySQL," has been streamlined and updated to MySQL 5.7.
- Microsoft Windows Server 2016 is the server operating system, and Windows 10 is the workstation operating system generally discussed and illustrated in the text. These are the current Microsoft server and workstation operating systems.

- We have updated online Appendix H, "Getting Started with Web Servers, PHP and the NetBeans IDE" to cover current versions of the software. We are now using the NetBeans IDE instead of the Eclipse PDT IDE. This provides a better development environment with a much simpler set of product installations because the Java JDK and NetBeans are installed in one combined installation. This new material provides a simplified (but still detailed) introduction to the installation and use of the Microsoft IIS Web server, PHP, the Java JDK, and the NetBeans in Appendix H. All of these tools are then used for Web database-application development as discussed in Chapter 11.
- More topics related to physical database design are now covered in Appendix G, which has been retitled "Physical Database Design and Data Structures for Database Processing." Specifically, coverage of multicolumn index creation and use, clustering, and decomposition have been added to accompany the existing topics of file organizations and single-column indexes.
- The old Appendix J, "Business Intelligence Systems," and K, "Big Data," expanded on some of the topics in Chapter 12. All that material remains, but it has been added to and reorganized. Appendix J, "Business Intelligence Systems," now includes coverage and examples of decision trees. The old Appendix K, "Big Data," has been split into three appendices: Appendix K, "Big Data," introduces Big Data technologies. It now includes a section on creating and using a relational cloud database and provides context for Appendix I, "XML," and Appendix L, "JSON and Document Databases." Appendix L describes the JSON document model in detail and covers the installation and use (creation and retrieval of data) of a document DBMS (ArangoDB), as well as use of a cloud-based document DBMS (Microsoft Cosmos DB, formerly called Microsoft Azure DocumentDB).

Fundamentals, Design, and Implementation

With today's technology, it is impossible to utilize a DBMS successfully without first learning fundamental concepts. After years of developing databases with business users, we have developed what we believe to be a set of essential database concepts. These are augmented by the concepts necessitated by the increasing use of the Internet, the World Wide Web, and commonly available analysis tools. Thus, the organization and topic selection of the 15th edition are designed to:

- Present an early introduction to SQL queries.
- Use a "spiral approach" to database design.
- Use a consistent, generic Information Engineering (IE) Crow's Foot E-R diagram notation for data modeling and database design.
- Provide a detailed discussion of specific normal forms within a discussion of normalization that focuses on pragmatic normalization techniques.
- Use current DBMS technology: Microsoft Access 2016, Microsoft SQL Server 2017, Oracle Database 12*c* Release 2 (and alternatively Oracle Database Express Edition 11*g* Release 2), and MySQL 5.7.
- Create Web database applications based on widely used Web development technology.
- Provide an introduction to business intelligence (BI) systems.
- Discuss the dimensional database concepts used in database designs for data warehouses and online analytical processing (OLAP).
- Discuss the emerging and important topics of server virtualization, cloud computing, Big Data, and the NoSQL (Not only SQL) movement.

These changes have been made because it has become obvious that the basic structure of the earlier editions (up to and including the 9th edition–the 10th edition introduced many of the changes we used in the 11th, 12th, 13th, and 14th editions and retain in the 15th edition) was designed for a teaching environment that no longer exists. The structural changes to the book were made for several reasons:

- Unlike the early years of database processing, today's students have ready access to data modeling and DBMS products.
- Today's students are too impatient to start a class with lengthy conceptual discussions on data modeling and database design. They want to do something, see a result, and obtain feedback.
- In the current economy, students need to reassure themselves that they are learning marketable skills.

Early Introduction of SQL DML

Given these changes in the classroom environment, this book provides an early introduction to SQL data manipulation language (DML) SELECT statements. The discussion of SQL data definition language (DDL) and additional DML statements occurs in Chapters 7 and 8. By encountering SQL SELECT statements in Chapter 2, students learn early in the class how to query data and obtain results, seeing firsthand some of the ways that database technology will be useful to them.

The text assumes that students will work through the SQL statements and examples with a DBMS product. This is practical today because nearly every student has access to Microsoft Access. Therefore, Chapters 1 and 2 and Appendix A–Getting Started with Microsoft Access 2016, are written to support an early introduction of Microsoft Access 2016 and the use of Microsoft Access 2016 for SQL queries (Microsoft Access 2016 QBE query techniques are also covered).

If a non-Microsoft Access-based approach is desired, versions of Microsoft SQL Server 2017, Oracle Database, and MySQL 5.7 are readily available for use. Free versions of the three major DBMS products covered in this book (SQL Server 2017 Developer Edition; Oracle Database Express Edition 11g Release 2 [Oracle Database XE], and MySQL 5.7 Community Edition) are available for download. Thus, students can actively use a DBMS product by the end of the first week of class.

BY THE WAY The presentation and discussion of SQL are spread over four chapters so students can learn about this important topic in small bites. SQL SELECT statements are taught in Chapter 2. SQL data definition language (DDL) and SQL data manipulation language (DML) statements are presented in Chapter 7. Correlated subqueries and EXISTS/NOT EXISTS statements are described in Chapter 8, and SQL transaction control language (TCL) and SQL data control language (DCL) are discussed in Chapter 9. Each topic appears in the context of accomplishing practical tasks. Correlated subqueries, for example, are used to verify functional dependency assumptions, a necessary task for database redesign.

This box illustrates another feature used in this book: BY THE WAY boxes are used to separate comments from the text discussion. Sometimes they present ancillary material; other times they reinforce important concepts.

A Spiral Approach to the Database Design Process

Today, databases arise from three sources: (1) from the need to integrate existing data from spreadsheets, data files, and database extracts; (2) from the need to develop new information systems projects; and (3) from the need to redesign an existing database to adapt to changing requirements. We believe that the fact that these three sources exist presents instructors with a significant pedagogical opportunity. Rather than teach database design just once from data models, why not teach database design three times, once for each of these sources? In practice, this idea has turned out to be even more successful than expected.

Database Design Iteration 1: Databases from Existing Data

Considering the design of databases from existing data, if someone were to email us a set of tables and say, "Create a database from them," how would we proceed? We would examine the tables in light of normalization criteria and then determine whether the new database was for a production system that allows new data to be inserted for each new transaction, or for a business intelligence (BI) data warehouse that allow users to only query data for use in reports and data analysis. Depending on the answer, we would normalize the data, pulling them apart (for the production transaction processing system), or denormalize the data, joining them together (for the BI system data warehouse). All of this is important for students to know and understand.

Therefore, the first iteration of database design gives instructors a rich opportunity to teach normalization, not as a set of theoretical concepts, but rather as a useful toolkit for making design decisions for databases created from existing data. Additionally, the construction of databases from existing data is an increasingly common task that is often assigned to junior staff members. Learning how to apply normalization to the design of databases from existing data not only provides an interesting way of teaching normalization, it is also common and useful!

We prefer to teach and use a pragmatic approach to normalization and present this approach in Chapter 3. However, we are aware that many instructors like to teach normalization in the context of a step-by-step normal form presentation (1NF, 2NF, 3NF, then BCNF), and Chapter 3 now includes additional material to provide more support for this approach as well.

In today's workplace, large organizations are increasingly licensing standardized software from vendors such as SAP, Oracle, and Siebel. Such software already has a database design. But with every organization running the same software, many are learning that they can gain a competitive advantage only if they make better use of the data in those predesigned databases. Hence, students who know how to extract data and create read-only databases for reporting and data mining have obtained marketable skills in the world of ERP and other packaged software solutions.

Database Design Iteration 2: Data Modeling and Database Design

The second source of databases is from new systems development. Although not as common as in the past, many databases are still created from scratch. Thus, students still need to learn data modeling, and they still need to learn how to transform data models into database designs that are then implemented in a DBMS product.

The IE Crow's Foot Model as a Design Standard

This edition uses a generic, standard IE Crow's Foot notation. Your students should have no trouble understanding the symbols and using the data modeling or database design tool of your choice.

IDEF1X (which was used as the preferred E-R diagram notation in the ninth edition of this text) is explained in Appendix C, "E-R Diagrams and the IDEF1X and UML Standards," in case your students will graduate into an environment where it is used or if you prefer to use it in your classes. UML is also explained in this appendix in case you prefer to use UML in your classes.

BY THE WAY The choice of a data modeling tool is somewhat problematic. Of the two most readily available tools, Microsoft Visio 2016 has been rewritten as a very rudimentary database design tool, whereas Oracle's MySQL Workbench is a database design tool, not a data modeling tool. MySQL Workbench cannot produce an N:M relationship as such (as a data model requires) but has to immediately break it into two 1:N relationships (as database design does). Therefore, the intersection table must be constructed and modeled. This confounds data modeling with database design in just the way that we are attempting to teach students to avoid.

To be fair to Microsoft Visio 2016, it is true that data models with N:M relationships can be drawn using the standard Microsoft Visio 2016 drawing tools. Unfortunately, Microsoft has chosen to remove many of the best database design tools that were in Microsoft Visio 2010, and Microsoft Visio 2016 lacks the tools that made it a favorite of Microsoft Access and Microsoft SQL Server users. For a full discussion of these tools, see Appendix D, "Getting Started with Microsoft Visio 2016.", and Appendix E, "Getting Started with the MySQL Workbench Data Modeling Tools".

Good data modeling tools are available, but they tend to be more complex and expensive. Two examples are Visible Systems' Visible Analyst and erwin Inc.'s erwin Data Modeler. Visible Analyst is available in a student edition (at a modest price), and a free trial period is available for erwin Data Modeler.

Database Design from E-R Data Models

As we discuss in Chapter 6, designing a database from data models consists of three tasks: representing entities and attributes with tables and columns; representing maximum cardinality by creating and placing foreign keys; and representing minimum cardinality via constraints, triggers, and application logic.

The first two tasks are straightforward. However, designs for minimum cardinality are more difficult. Required parents are easily enforced using NOT NULL foreign keys and referential integrity constraints. Required children are more problematic. In this book, however, we simplify the discussion of this topic by limiting the use of referential integrity actions and by supplementing those actions with design documentation. See the discussion around Figure 6-29.

Although the design for required children is complicated, it is important for students to learn. It also provides a reason for students to learn about triggers as well. In any case, the discussion of these topics is much simpler than it was in prior editions because of the use of the IE Crow's Foot model and ancillary design documentation.

Database Implementation from Database Designs

Of course, to complete the process, a database design must be implemented in a DBMS product. This is discussed in Chapter 7, where we introduce SQL DDL for creating tables and SQL DML for populating the tables with data.

BY THE WAY David Kroenke is the creator of the semantic object model (SOM). The SOM is presented in Appendix F, "The Semantic Object Model." The E-R data model is used everywhere else in the text.

Database Design Iteration 3: Database Redesign

Database redesign, the third iteration of database design, is both common and difficult. As stated in Chapter 8, information systems cause organizational change. New information systems give users new behaviors, and as users behave in new ways, they require changes in their information systems.

Database redesign is, by nature, complex. Depending on your students, you may wish to skip it, and you can do so without loss of continuity. Database redesign is presented after the discussion of SQL DDL and DML in Chapter 7 because it requires the use of advanced SQL. It also provides a practical reason to teach correlated subqueries and EXISTS/NOT EXISTS statements.

Active Use of a DBMS Product

We assume that students will actively use a DBMS product. The only real question becomes "which one?" Realistically, most of us have four alternatives to consider: Microsoft Access, Microsoft SQL Server, Oracle Database, and MySQL. You can use any of those products with this text, and tutorials for each of them are presented for Microsoft Access 2016 (Appendix A), SQL Server 2017 (Chapter 10A), Oracle Database 12*c* Release 2 and Oracle Database XE (Chapter 10B), and MySQL 5.7 (Chapter 10C). Given the limitations of class time, it is probably necessary to pick and use just one of these products. You can often devote a portion of a lecture to discussing the characteristics of each, but it is usually best to limit student work to one of them. The possible exception to this is starting the course with Microsoft Access and then switching to a more robust DBMS product later in the course.

Using Microsoft Access 2016

The primary advantage of Microsoft Access is accessibility. Most students already have a copy, and, if not, copies are easily obtained. Many students will have used Microsoft Access in their introductory or other classes. Appendix A, "Getting Started with Microsoft Access 2016, " is a tutorial on Microsoft Access 2016 for students who have not used it but who wish to use it with this book.

However, Microsoft Access has several disadvantages. First, as explained in Chapter 1, Microsoft Access is a combination application generator and DBMS. Microsoft Access confuses students because it confounds database processing with application development. Also, Microsoft Access 2016 hides SQL behind its query processor and makes SQL appear as an afterthought rather than a foundation. Furthermore, as discussed in Chapter 2, Microsoft Access 2016 does not correctly process some of the basic SQL-92 standard statements in its default setup. Finally, Microsoft Access 2016 does not support triggers. You can simulate triggers by trapping Windows events, but that technique is nonstandard and does not effectively communicate the nature of trigger processing.

Using Microsoft SQL Server 2017, Oracle Database, or MySQL 5.7

Choosing which of these products to use depends on your local situation. Oracle Database 12*c* Release 2, a superb enterprise-class DBMS product, is difficult to install and administer. However, if you have local staff to support your students, it can be an excellent choice. Fortunately, Oracle Database Express Edition 11*g* Release 2, commonly referred to as Oracle Database XE, is easy to install, easy to use, and freely downloadable. If you want your students to be able to install Oracle Database on their own computers, use Oracle Database XE. As shown in Chapter 10B, Oracle's SQL Developer GUI tool (or SQL*Plus if you are dedicated to this beloved command-line tool) is a handy tool for learning SQL, triggers, and stored procedures.

Microsoft SQL Server 2017, although probably not as robust as Oracle Database, is easy to install on Windows machines, and it provides the capabilities of an enterprise-class DBMS product. The standard database administrator tool is the Microsoft SQL Server Management Studio GUI tool. As shown in Chapter 10A, SQL Server 2017 can be used to learn SQL, triggers, and stored procedures.

MySQL 5.7, discussed in Chapter 10C, is an open source DBMS product that is receiving increased attention and market share. The capabilities of MySQL are continually being upgraded, and MySQL 5.7 supports stored procedures and triggers. MySQL also has excellent GUI tools in the MySQL Workbench and an excellent command-line tool (the MySQL Command Line Client). It is the easiest of the three products for students to install on their own computers. It also works with the Linux operating system and is popular as part of the AMP (Apache–MySQL–PHP) package (known as WAMP on Windows and LAMP on Linux).

BY THE WAY Because we only present currently available software products in this book, we cover MySQL 5.7. However, MySQL 8.0 is currently in development status, which means that it will be generally available in the near future.

BY THE WAY If the DBMS you use is not driven by local circumstances and you do have a choice, we recommend using Microsoft SQL Server 2017. It has all of the features of an enterprise-class DBMS product, and it is easy to install and use. Another option is to start with Microsoft Access 2016 if it is available and switch to SQL Server 2017 at Chapter 7. Chapters 1 and 2 and Appendix A are written specifically to support this approach. A variant is to use Microsoft Access 2016 as the development tool for forms and reports running against an SQL Server 2017 database.

If you prefer a different DBMS product, you can still start with Microsoft Access 2016 and switch later in the course. See the detailed discussion of the available DBMS products in Chapter 10 for a good review of your options.

Focus on Database Application Processing

In this edition, we clearly draw the line between *application development* per se and *database application processing*. Specifically, we have:

- Focused on specific database-dependent applications:
  - Web-based, database-driven applications

  - XML-based data processing
  - Business intelligence (BI) systems applications
- Emphasized the use of commonly available, multiple-OS-compatible application development languages.
- Limited the use of specialized vendor-specific tools and programming languages as much as possible.

There is simply not enough room in this book to provide even a basic introduction to programming languages used for application development such as the Microsoft .NET languages and Java. Therefore, rather than attempting to introduce these languages, we leave them for other classes where they can be covered at an appropriate depth. Instead, we focus on basic tools that are relatively straightforward to learn and immediately applicable to database-driven applications. We use PHP as our Web development language, and we use the readily available NetBeans integrated development environment (IDE) as our development tool. The result is a *very* focused final section of the book, where we deal specifically with the interface between databases and the applications that use them.

BY THE WAY Although we try to use widely available software as much as possible, there are, of course, exceptions where we must use vendor-specific tools. For BI applications, for example, we draw on Microsoft Excel's PivotTable capabilities and the Microsoft PowerPivot for Microsoft Excel 2016 add-in. However, either alternatives to these tools are available (OpenOffice.org DataPilot capabilities, the Palo OLAP Server) or the tools are generally available for download.

Business Intelligence Systems and Dimensional Databases

This edition maintains coverage of business intelligence (BI) systems (Chapter 12 and Appendix J). The chapter includes a discussion of dimensional databases, which are the underlying structure for data warehouses, data marts, and OLAP servers. It still covers data management for data warehouses and data marts and also describes reporting and data mining applications, including OLAP.

Appendix J includes in-depth coverage of three applications that should be particularly interesting to students. The first is RFM analysis, a reporting application frequently used by mail order and e-commerce companies. The complete RFM analysis is accomplished in Appendix J through the use of standard SQL statements. The second, market basket analysis, is used by organizations to find patterns in purchase (or similar) data. Decision trees, the third topic covered in depth in Appendix J, are used to automatically categorize records based on past experience (e.g., is a customer a high or low risk for insurance coverage?). Appendix J can be assigned at any point after Chapter 8 and could be used as a motivator to illustrate the practical applications of SQL midcourse. Finally, Appendix K and Appendix L provide additional material on Big Data and NoSQL databases to supplement and support Chapter 12.

Overview of the Chapters in the 15th Edition

Chapter 1 sets the stage by introducing database processing, describing basic components of database systems, and summarizing the history of database processing. If students are using Microsoft Access 2016 for the first time (or need a good review), they will also need to study Appendix A, "Getting Started with Microsoft Access 2016" at this point. Chapter 2 presents SQL SELECT statements. It also includes sections on how to submit SQL statements to Microsoft Access 2016, SQL Server 2017, Oracle Database, and MySQL 5.7.

The next four chapters, Chapters 3 through 6, present the first two iterations of database design. Chapter 3 presents the principles of normalization to Boyce-Codd Normal Form (BCNF). It describes the problems of multivalued dependencies and explains how to eliminate them. This foundation in normalization is applied in Chapter 4 to the design of databases from existing data.

Chapters 5 and 6 describe the design of new databases. Chapter 5 presents the E-R data model. Traditional ER symbols are explained, but the majority of the chapter uses IE Crow's Foot notation. Chapter 5 provides a taxonomy of entity types, including strong, ID-dependent, weak but not ID-dependent, supertype/subtype, and recursive. The chapter concludes with a simple modeling example for a university database.

Chapter 6 describes the transformation of data models into database designs by converting entities and attributes to tables and columns; by representing maximum cardinality by creating and placing foreign keys; and by representing minimum cardinality via carefully designed DBMS constraints, triggers, and application code. The primary section of this chapter parallels the entity taxonomy in Chapter 5.

Chapter 7 presents SQL DDL, DML, and SQL/Persistent Stored Modules (SQL/PSM). SQL DDL is used to implement the design of an example introduced in Chapter 6. INSERT, UPDATE, MERGE, and DELETE statements are discussed, as are SQL views. Additionally, the principles of embedding SQL in program code are presented, SQL/PSM is discussed, and triggers and stored procedures are explained.

Database redesign, the third iteration of database design, is described in Chapter 8. This chapter presents SQL statements using correlated subqueries and the SQL EXIST and NOT EXISTS operators, and uses these statements in the redesign process. Reverse engineering is described, and basic redesign patterns are illustrated and discussed.

Chapters 9, 10, 10A, 10B, and 10C consider the management of multiuser organizational databases. Chapter 9 describes database administration tasks, including concurrency, security, and backup and recovery. Chapter 10 is a general introduction to the online Chapters 10A, 10B, and 10C, which describe SQL Server 2017, Oracle Database (both Oracle Database 12*c* Release 2 and Oracle Database XE), and MySQL 5.7, respectively. These chapters show how to use these specific products to create database structures and process SQL statements. They also explain concurrency, security, and backup and recovery with each product. The discussion in Chapters 10A, 10B, and 10C parallels the order of discussion in Chapter 9 as much as possible, though rearrangements of some topics are made, as needed, to support the discussion of a specific DBMS product.

BY THE WAY We have maintained or extended our coverage of Microsoft Access, Microsoft SQL Server, Oracle Database, and MySQL (introduced in *Database Processing: Fundamentals, Design, and Implementation,* 11th edition) in this book. In order to keep the bound book to a reasonable length and to keep the cost of the book down, we have chosen to provide some material by download from our Web site at *www.pearsonhighered.com/kroenke*. There you will find:

- *Chapter 10A—Managing Databases with Microsoft SQL Server 2017*
- *Chapter 10B—Managing Databases with Oracle Database*
- *Chapter 10C—Managing Databases with MySQL 5.7*
- *Appendix A—Getting Started with Microsoft Access 2016*
- *Appendix B—Getting Started with Systems Analysis and Design*
- *Appendix C—E-R Diagrams and the IDEF1X and UML Standards*
- *Appendix D— Getting Started with Microsoft Visio 2016*
- *Appendix E—Getting Started with the MySQL Workbench Data Modeling Tools*
- *Appendix F— The Semantic Object Model*
- *Appendix G—Physical Database Design and Data Structures for Database Processing*
- *Appendix H—Getting Started with Web Servers, PHP, and the NetBeans IDE*
- *Appendix I—XML*
- *Appendix J—Business Intelligence Systems*
- *Appendix K—Big Data*
- *Appendix L—JSON and Document Databases*

Chapters 11 and 12 address standards for accessing databases. Chapter 11 presents ODBC, OLE DB, ADO.NET, ASP.NET, JDBC, and JavaServer Pages (JSP). It then introduces PHP (and the NetBeans IDE) and illustrates the use of PHP for the publication of databases via Web pages. This is followed by a description of the integration of XML and database technology. The chapter begins with a primer on XML and then shows how to use the FOR XML SQL statement in SQL Server.

Chapter 12 concludes the text with a discussion of BI systems, dimensional data models, data warehouses, data marts, server virtualization, cloud computing, Big Data, structured storage, and the Not only SQL movement.

Supplements

This text is accompanied by a wide variety of supplements. Please visit the text's Web site at *www.pearsonhighered.com/kroenke* to access the instructor and student supplements described next. Please contact your Pearson sales representative for more details. All supplements were written by David Auer, Scott Vandenberg, Bob Yoder, and Harold Wise.

For Students

Many of the sample databases used in this text are available online in Microsoft Access, Microsoft SQL Server 2017, Oracle Database, and MySQL 5.7 formats.

For Instructors

At the Instructor Resource Center, *www.pearsonhighered.com/irc*, instructors can access a variety of print, digital, and presentation resources available with this text in downloadable format. Registration is simple and gives instructors immediate access to new titles and new editions. As a registered faculty member, you can download resource files and receive immediate access to and instructions for installing course management content on your campus server. In case you ever need assistance, our dedicated technical support team is ready to help with the media supplements that accompany this text. Visit *http://247.pearsoned.com* for answers to frequently asked questions and toll-free user support phone numbers.

The following supplements are available for download to adopting instructors:

- Instructor's Manual (including database files and solutions)
- Test Bank
- TestGen Computerized Test Bank
- PowerPoint Presentations

Acknowledgments

We are grateful for the support of many people in the development of this 15th edition and previous editions. Kraig Pencil of Western Washington University helped us refine the use of the book in the classroom. Recently David Auer and Xiaofeng Chen team-taught a database class together at Western Washington University, and our interaction and discussions with Professor Chen resulted in several modifications and improvements in this book. Professor Chen also graciously allowed us to adopt some of his classroom examples for use in the books. Thanks are also due to Barry Flachsbart of Missouri University of Science and Technology and Don Malzahn of Harper College for their comments and SQL code checking. Finally, thanks to Donna Auer for giving us permission to use her painting *waterfall* as the cover art for this book.

In addition, we wish to thank the reviewers of this edition:

Brian Bender, *Northern Illinois University*
Larry Booth, *Clayton State University*
Richard Chrisman, *Northeast Community College*
Vance Cooney, *Eastern Washington University*

Kui Du, *University of Massachusetts Boston*
John N Dyer, *Georgia Southern University*
Richard Egan, *New Jersey Institute of Technology*
David Fickbohm, *Golden Gate University*
Edward Garrity, *Canisius College*
Mary Jo Geise, *The University of Findlay*
Richard Goeke, *Widener University*
Pranshu Gupta, *DeSales University*
Reggie Haseltine, *CSU Global*
Gerald Hensel, *Valencia College*
Carole Hollingsworth, *Kennesaw State University*
Simon Jin, *Metropolitan State University*
Darrell Karbginsky, *Chemeketa Community College*
Stephen Larson, *Slippery Rock University of PA*
Taowen Le, *Weber State University*
Chang Liu, *Northern Illinois University*
Nicole Lytle-Kosola, *University of La Verne*
Parand Mansouri Rad, *CSU Chico*
Chris Markson, *New Jersey Institute of Technology*
Vishal Midha, *Illinois State University*
Atreyee Sinha, *Edgewood College*
Todd Will, *New Jersey Institute of Technology*
Russ Wright, *College of Central Florida*

Finally, we would like to thank Samantha Lewis, our Portfolio Manager; Stephany Harrington, our Content Producer; Revathi Viswanathan, our Project Manager; and Mirasol Dante, our Production Liason; for their professionalism, insight, support, and assistance in the development of this project. We would also like to thank Harold Wise of East Carolina University for his detailed comments on the final manuscript—this book would not be what it is without their extensive input. Finally, David Kroenke would like to thank his wife, Lynda; David Auer would like to thank his wife, Donna; Scott Vandenberg would like to thank his wife, Kristin; and Robert Yoder would like to thank Diane, Rachael, and Harrison for their love, encouragement, and patience while this project was being completed. David Kroenke would further like to thank David Auer for keeping this book going, and Scott Vandenberg and Robert Yoder for their contributions as the new members of the team!

David Kroenke
Whidbey Island, Washington

David Auer
Bellingham, Washington

Scott Vandenberg
Loudonville, New York

Robert Yoder
Loudonville, New York

About the Authors

David M. Kroenke

Work Experience

David M. Kroenke has more than 50 years of experience in the computer industry. He began as a computer programmer for the U.S. Air Force, working both in Los Angeles and at the Pentagon, where he developed one of the world's first DBMS products while part of a team that created a computer simulation of World War III. That simulation served a key role for strategic weapons studies during a 10-year period of the Cold War.

From 1973 to 1978, Kroenke taught in the College of Business at Colorado State University. In 1977 he published the first edition of *Database Processing*, a significant and successful textbook that, more than 40 years later, you now are reading in its 15th edition. In 1978, he left Colorado State and joined Boeing Computer Services, where he managed the team that designed database management components of the IPAD project. After that, he joined with Steve Mitchell to form Mitchell Publishing and worked as an editor and author, developing texts, videos, and other educational products and seminars. Mitchell Publishing was acquired by Random House in 1986. During those years, he also worked as an independent consultant, primarily as a database disaster repairman helping companies recover from failed database projects.

In 1982, Kroenke was one of the founding directors of the Microrim Corporation. From 1984 to 1987, he served as the Vice President of Product Marketing and Development and managed the team that created and marketed the DBMS product RBASE 5000 as well as other related products.

For the next five years, Kroenke worked independently while he developed a new data modeling language called the *semantic object model*. He licensed this technology to the Wall Data Corporation in 1992 and then served as the Chief Technologist for Wall Data's Salsa line of products. He was awarded three software patents on this technology.

Since 1998, Kroenke has continued consulting and writing. His current interests concern the practical applications of data mining techniques on large organizational databases. An avid sailor, he wrote *Know Your Boat: The Guide to Everything That Makes Your Boat Work*, which was published by McGraw-Hill in 2002.

Consulting

Kroenke has consulted with numerous organizations during his career. In 1978, he worked for Fred Brooks, consulting with IBM on a project that became the DBMS product DB2. In 1989, he consulted for the Microsoft Corporation on a project that became Microsoft Access. In the 1990s, he worked with Computer Sciences Corporation and with General Research Corporation for the development of technology and products that were used to model all of the U.S. Army's logistical data as part of the CALS project. Additionally, he has consulted for Boeing Computer Services, the U.S. Air Force Academy, Logicon Corporation, and other smaller organizations.

Publications

- *Database Processing*, Pearson Prentice Hall, 15 editions, 1977–present (coauthor with David Auer, 11th, 12th, 13th, and 14th editions. Coauthor with David Auer, Scott Vandenberg, and Robert Yoder 15th edition.)

- *Database Concepts*, Pearson Prentice Hall, eight editions, 2004–present (coauthor with David Auer, 3rd, 4th, 5th, 6th, and 7th editions. Coauthor with David Auer, Scott Vandenberg, and Robert Yoder 8th edition.)
- *Using MIS*, Pearson Prentice Hall, ten editions, 2006–present (coauthor with Randall J. Boyle, 8th, 9th, and 10th edition)
- *Experiencing MIS*, Pearson Prentice Hall, six editions, 2007–present (coauthor with Randall J. Boyle, 6th edition)
- *MIS Essentials*, Pearson Prentice Hall, four editions, 2009–present
- *Processes, Systems, and Information: An Introduction to MIS*, Pearson Prentice Hall, two editions, 2013–present (coauthor with Earl McKinney)
- *Essentials of Processes, Systems, and Information*, Pearson Prentice Hall, 2013 (coauthor with Earl McKinney)
- *Know Your Boat: The Guide to Everything That Makes Your Boat Work*, McGraw-Hill, 2002
- *Management Information Systems*, Mitchell Publishing/Random House, three editions, 1987–1992
- *Business Computer Systems*, Mitchell Publishing/Random House, five editions, 1981–1990
- *Managing Information for Microcomputers*, Microrim Corporation, 1984 (coauthor with Donald Nilson)
- *Database Processing for Microcomputers*, Science Research Associates, 1985 (coauthor with Donald Nilson)
- *Database: A Professional's Primer*, Science Research Associates, 1978

Teaching

Kroenke taught in the College of Business at Colorado State University from 1973 to 1978. He also has taught part time in the Software Engineering program at Seattle University. From 1990 to 1991, he served as the Hanson Professor of Management Science at the University of Washington. Most recently, he taught at the University of Washington from 2002 to 2008. During his career, he has been a frequent speaker at conferences and seminars for computer educators. In 1991, the International Association of Information Systems named him Computer Educator of the Year.

Education

B.S., Economics, U.S. Air Force Academy, 1968
M.S., Quantitative Business Analysis, University of Southern California, 1971
Ph.D., Engineering, Colorado State University, 1977

Personal

Kroenke is married, lives in Seattle, and has two grown children and three grandchildren. He enjoys skiing, sailing, and building small boats. His wife tells him he enjoys gardening as well.

David J. Auer

Work Experience

David J. Auer has more than 30 years of experience teaching college-level business and information systems courses and for the past 20 years has worked professionally in the field of information technology. He served as a commissioned officer in the U.S. Air Force, with assignments to NORAD and the Alaskan Air Command in air defense operations. He later taught both business administration and music classes at Whatcom Community College and business courses for the Chapman College Residence Education Center at Whidbey Island Naval Air Station. He was a founder of the Puget Sound Guitar Workshop (now in its 41st year of operations). He worked as a psychotherapist and organizational development consultant for the Whatcom Counseling and Psychiatric Clinic's Employee Assistance Program and provided training for the Washington State Department of Social and Health

Services. He taught for Western Washington University's College of Business and Economics from 1981 to June 2015 and served as the college's Director of Information Systems and Technology Services from 1994 to 2014. Now a Senior Instructor Emeritus at Western Washington University, he continues his writing projects.

Publications

- *Database Processing*, Pearson Prentice Hall, five editions, 2009–present (coauthor with David Kroenke, eighth edition coauthor with David Kroenke, Scott L. Vandenberg, and Robert C. Yoder)
- *Database Concepts*, Pearson Prentice Hall, six editions, 2007–present (coauthor with David Kroenke, eighth edition coauthor with David Kroenke, Scott L. Vandenberg, and Robert C. Yoder)
- *Network Administrator: NetWare 4.1*, Course Technology, 1997 (coauthor with Ted Simpson and Mark Ciampa)
- *New Perspectives on Corel Quattro Pro 7.0 for Windows 95*, Course Technology, 1997 (coauthor with June Jamrich Parsons, Dan Oja, and John Leschke)
- *New Perspectives on Microsoft Excel 7 for Windows 95–Comprehensive*, Course Technology, 1996 (coauthor with June Jamrich Parsons and Dan Oja)
- *New Perspectives on Microsoft Office Professional for Windows 95–Intermediate*, Course Technology, 1996 (coauthor with June Jamrich Parsons, Dan Oja, Beverly Zimmerman, Scott Zimmerman, and Joseph Adamski)
- *Microsoft Excel 5 for Windows–New Perspectives Comprehensive*, Course Technology, 1995 (coauthor with June Jamrich Parsons and Dan Oja)
- *Introductory Quattro Pro 6.0 for Windows*, Course Technology, 1995 (coauthor with June Jamrich Parsons and Dan Oja)
- *Introductory Quattro Pro 5.0 for Windows*, Course Technology, 1994 (coauthor with June Jamrich Parsons and Dan Oja)
- *The Student's Companion for Use with Practical Business Statistics*, Irwin, two editions 1991 and 1993

Teaching

Auer taught in the College of Business and Economics at Western Washington University from 1981 to June 2015. From 1975 to 1981, he taught part time for community colleges, and from 1981 to 1984, he taught part time for the Chapman College Residence Education Center System. During his career, he has taught a wide range of courses in Quantitative Methods, Production and Operations Management, Statistics, Finance, and Management Information Systems. In MIS, he has taught Principles of Management Information Systems, Business Database Development, Computer Hardware and Operating Systems, Telecommunications, Network Administration, and Fundamentals of Web Site Development.

Education

B.A., English Literature, University of Washington, 1969
B.S., Mathematics and Economics, Western Washington University, 1978
M.A., Economics, Western Washington University, 1980
M.S., Counseling Psychology, Western Washington University, 1991

Personal

Auer is married, lives in Bellingham, Washington, and has two grown children and four grandchildren. He is active in his community, where he has been president of his neighborhood association and served on the City of Bellingham Planning and Development Commission. He enjoys music, playing acoustic and electric guitar, five-string banjo, and a bit of mandolin.

Scott L. Vandenberg

Work Experience

Scott L. Vandenberg has over 25 years' experience teaching computer science to college students in computer science and business. Before completing his PhD, he worked for brief periods at Standard Oil Research, Procter & Gamble headquarters, and IBM Research. He taught for two years at the University of Massachusetts-Amherst before joining the faculty at Siena College in 1993. His main teaching interests are in the areas of database management systems and introductory computer science, with research, consulting, and publications focused on those areas as well. Some of his earlier scholarly work included development of data models, query languages, and algebras for object-oriented databases and databases involving sequential and tree-structured data. More recent research has involved applying database technology to help solve data science problems in the areas of biology and epidemiology. He has also published several papers relating to introductory computer science curricula and is currently a co-principal investigator on a multiyear NSF grant to develop methods to broaden participation and increase retention in computer science. Vandenberg has published over 20 papers related to his scholarly activity.

Publications

- *Database Concepts*, Pearson Prentice Hall, 8th edition, 2017 (coauthor with David Kroenke, David Auer, and Robert Yoder)
- *Database Processing*, Pearson Prentice Hall, 15th edition, 2018 (coauthor with David Kroenke, David Auer, and Robert Yoder)

Teaching

Vandenberg has been on the Computer Science faculty at Siena College since 1993, where he regularly teaches three different database courses at several levels to both Computer Science majors and Business majors. Prior to arriving at Siena, he taught undergraduate and graduate courses in database systems at the University of Massachusetts-Amherst. Since arriving at Siena, he also has taught graduate and undergraduate database courses at the University of Washington in Seattle. He has developed five different database courses over this time. His other teaching experience includes introductory computer science, introductory programming, data structures, management information systems, and three years teaching Siena's interdisciplinary freshman writing course.

Education

B.A., Computer Science and Mathematics, Cornell University, 1986
M.S., Computer Science, University of Wisconsin-Madison, 1987
Ph.D., Computer Science, University of Wisconsin-Madison, 1993

Personal

Vandenberg is married; lives in Averill Park, New York; and has two children. When not playing with databases, he enjoys playing ice hockey and studying medieval history.

Robert C. Yoder

Work Experience

Robert C. Yoder began his professional career at the University at Albany as a systems programmer managing Unisys and IBM mainframes, along with Unix servers. He became the Assistant Director of Systems Programming, gaining over 25 years' experience as a programmer and technical manager.

Bob took a two-year break from systems programming to work as a senior systems analyst at Phoenix Data Systems in Albany, New York. He assisted a team to develop an innovative 3-D solid modeling system using a data structure called octree encoding that can represent the interior properties of objects. This work became the inspiration for his PhD dissertation on 3-D geographic information systems.

Publications

- *Database Concepts*, Pearson Prentice Hall, 8th edition, 2017 (coauthor with David Kroenke, David Auer, and Scott Vandenberg)
- *Database Processing*, Pearson Prentice Hall, 15th edition, 2018 (coauthor with David Kroenke, David Auer, and Scott Vandenberg)

Teaching

Teaching is Bob's second career. He started teaching computer science courses as an adjunct at the University of Albany (SUNY) and Siena College, and then accepted full-time employment at Siena College's Computer Science Department in 2001. Bob teaches data structures, business database, operating systems, Java programming, geographic information systems, and management information systems. Bob has published several academic papers relating to management information systems, globalization, data structures, and computer science education.

Education

B.S., Computer Science and Applied Mathematics, University at Albany, 1977
M.S., Computer Science, University at Albany, 1979
Ph.D., Information Science, University at Albany, 1999

Personal

Bob lives in Niskayuna, New York, with his wife Diane and has two children. He enjoys traveling, hiking, and walking his dog. Bob would like to dedicate his portion of the textbook to the memory of loved ones who passed away recently: Dorothy Yoder, Laurie Gorski, and canine companion Robbie.

This page intentionally left blank

PART 1

Getting Started

The two chapters in Part 1 provide an introduction to database processing. In Chapter 1, we discuss the importance of databases to support Internet Web applications and smartphone apps. We then consider the characteristics of databases and describe important database applications. Chapter 1 discusses the various database components, provides a survey of the knowledge you need to learn from this text, and summarizes the history of database processing.

You will start working with a database in Chapter 2 and use that database to learn how to use Structured Query Language (SQL), a database-processing language, to query database data. You will learn how to query both single and multiple tables. Together, these two chapters will give you a sense of what databases are and how they are processed.

1 Introduction

Chapter Objectives

- To understand the importance of databases in Internet Web applications and smartphone apps
- To understand the nature and characteristics of databases
- To survey some important and interesting database applications
- To gain a general understanding of tables and relationships
- To describe the components of a Microsoft Access database system and explain the functions they perform
- To describe the components of an enterprise-class database system and explain the functions they perform
- To define the term *database management system* (DBMS) and describe the functions of a DBMS
- To define the term *database* and describe what is contained within the database
- To define the term *metadata* and provide examples of metadata
- To define and understand database design from existing data
- To define and understand database design as new systems development
- To define and understand database redesign of an existing database
- To understand the history and development of database processing

This chapter discusses the importance of databases in the Internet world and then introduces database processing concepts. We will first consider the nature and characteristics of databases and then survey a number of important and interesting database applications. Next, we will describe the components of a database system and then, in general terms, describe how databases are designed. After that, we will survey the knowledge that you need to work with databases as an application developer or as a database administrator. Finally, we conclude this introduction with a brief history of database processing.

To really understand databases and database technology, you will need to actively use some database product. Fortunately, in today's computer environment, easily obtainable versions of most major database products are available, and we will make use of them. However, this chapter assumes a minimal knowledge of database use. It assumes that you have used a basic database product such as Microsoft Access to enter data into a form, to produce a report, and possibly to execute a query. If you have not done these things, you should obtain a copy of Microsoft Access 2016 and work through the tutorial in Appendix A.

The Importance of Databases in the Internet and Smartphone World

Let's stop for a moment and consider the incredible information technology available for our use today.

The **personal computer (PC)** became widely available with the introduction of the **Apple II** in 1977 and the **IBM Personal Computer (IBM PC)** in 1981. PCs were networked into **local area networks (LANs)** using the **Ethernet networking technology**, which was developed at the Xerox Palo Alto Research Center in the early 1970s and adopted as a national standard in 1983.

The **Internet**—the global computer network of networks—was created as the **ARPANET** in 1969 and then grew and was used to connect all the LANs (and other types of networks). The Internet became widely known and used when the **World Wide Web** (also referred to as **the Web** and **WWW**) became easily accessible in 1993. Everyone got a computer software application called a **Web browser** and starting *browsing* to **Web sites**. Online retail Web sites such as Amazon.com (online since 1995) and "brick-and-mortar" stores with an online presence such as Best Buy appeared, and people started extensively *shopping online*.

In the early 2000s, **Web 2.0**[1] Web sites started to appear—Web sites that allowed users to add content to Web sites that had previously held static content. Web applications such as Facebook, Wikipedia, and Twitter appeared and flourished.

In a parallel development, the **mobile phone** or **cell phone** was demonstrated and developed for commercial use in the 1970s. After decades of mobile phone and cell phone network infrastructure development, the **smartphone** appeared. Apple brought out the **iPhone** in 2007. Google created the **Android operating system**, and the first Android-based smartphone entered the market in 2008. Now, just a few years later, smartphones and **tablet computers (tablets)** have become widely used, and thousands of application programs known as **apps** are widely available and in daily use. Most Web applications now have corresponding smartphone and tablet apps (you can "tweet" from either your computer or your smartphone)!

What many people do not understand is that in today's Web application and smartphone app environment, most of what they do depends upon databases.

We can define **data** as recorded facts and numbers. We can initially define a *database* (we will give a better definition later in this chapter) as the structure used to hold or store that data. We process that data to provide *information* (which we also define in more detail later in this chapter) for use in the Web applications and smartphone apps.

Do you have a Facebook account? If so, all your posts, your comments, your "likes," and other data you provide to Facebook (such as photos) are stored in a *database*. When your friend posts an item, it is initially stored in the *database* and then displayed to you.

Do you have a Twitter account? If so, all your tweets are stored in a *database*. When your friend tweets something, it is initially stored in the *database* and then displayed to you.

Do you shop at Amazon.com? If so, how do you find what you are looking for? You enter some words in a Search text window on the Amazon home Web page (if you are using a Web browser) and click the Go button. Amazon's computers then search Amazon's *databases* and return a formatted report on screen of the items that matched what you searched for.

The search process is illustrated in Figure 1-1, where we search the Pearson Higher Education Web page for books authored by *David Kroenke*. Figure 1-1(a) shows the upper portion

[1] The term Web 2.0 was originated by Darcy DiNucci in 1999 and introduced to the world at large in 2004 by publisher Tim O'Reilly. See the Wikipedia article **Web 2.0** (accessed July 2017).

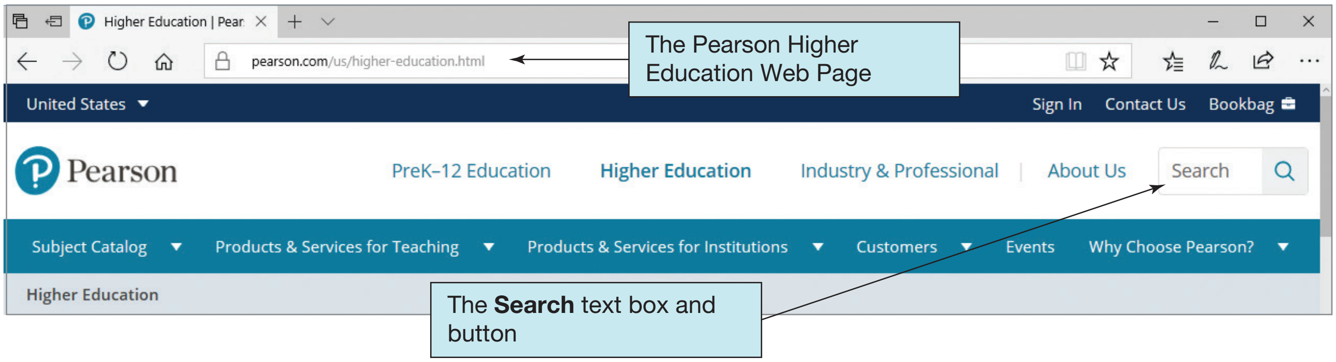
of the Pearson Higher Education Web page, with a Search text box and button in the upper right corner of the Web page. As shown in Figure 1-1(b), we enter the author name *Kroenke* in the text box and then click the **Search** button. The Pearson catalog database is searched, and the Web application returns a *Search Results Higher Education* page containing a listing of books authored by David Kroenke (appropriately starting with the listing for our companion book, *Database Concepts*), as shown in Figure 1-1(c).

BY THE WAY It is much more effective to see this process than to just read about it. Take a minute, open a Web browser, and go to Amazon.com (or any other online retailer, such as Best Buy, Crutchfield, or REI). Search for something you are interested in, and watch the database search results be displayed for you. You just used a *database*.

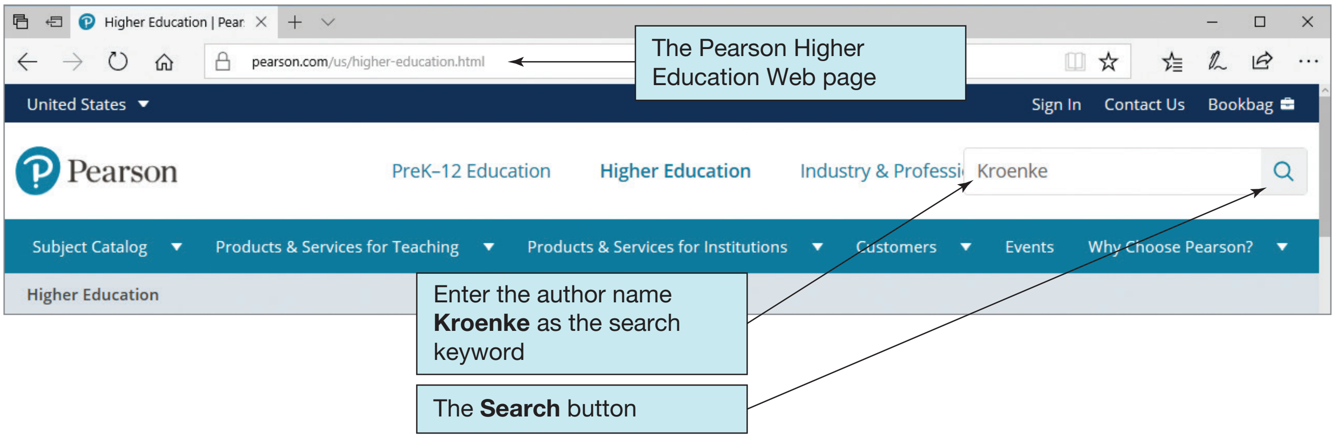
BY THE WAY Even if you are simply shopping in a local grocery store (or a coffee shop or pizzeria), you are interacting with databases. Businesses use **point of sale (POS) systems** to record every purchase in a database, to monitor inventory, and, if you have a sales promotion card from the store (the one you use to get those special prices for "card holders only"), to keep track of everything you buy for marketing purposes. All the data POS systems gather is stored in, of course, a *database*.

FIGURE 1-1

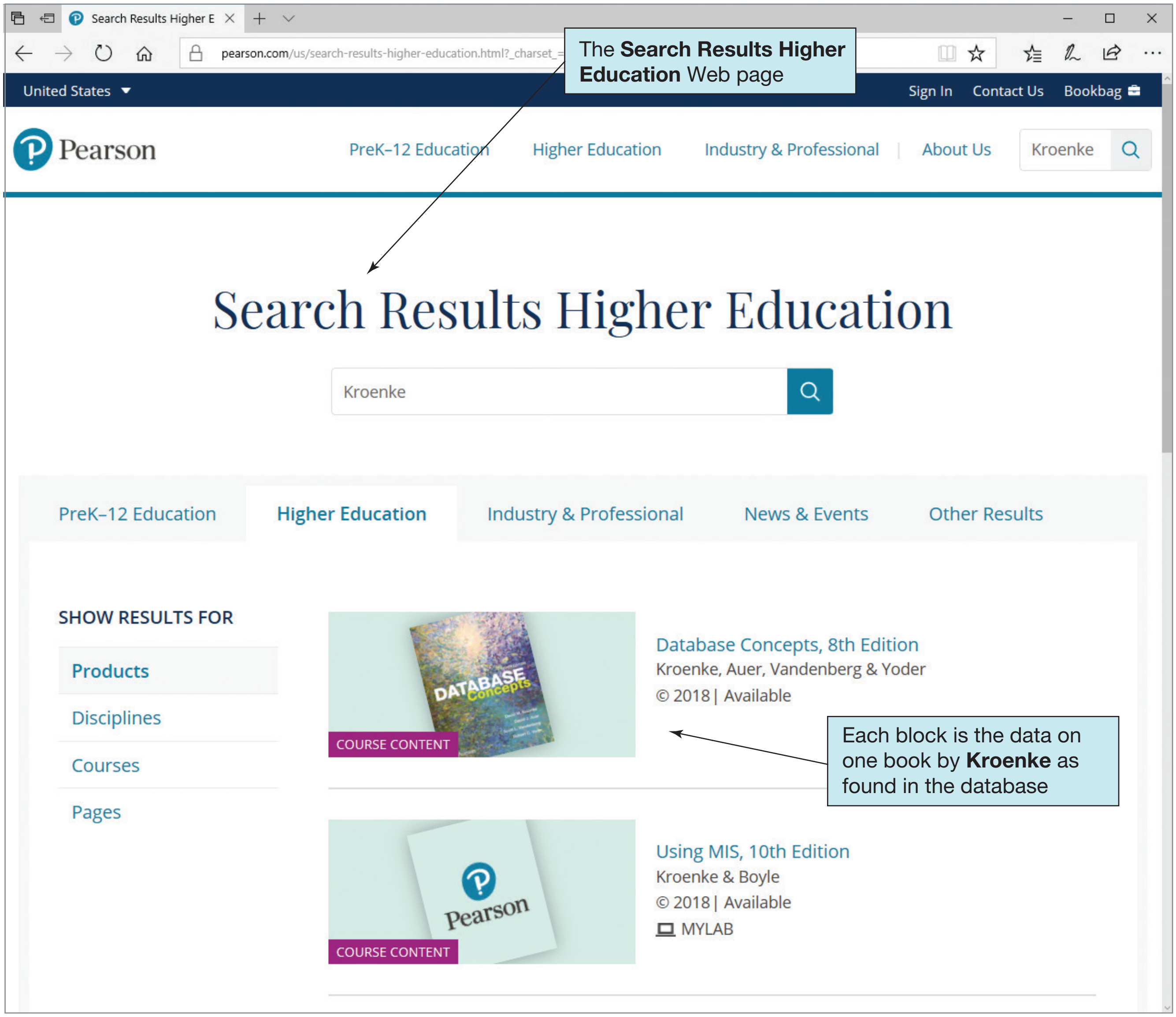
Searching a Database in a Web Browser



(a) The Pearson Higher Education Web Page



(b) Entering Author Name *Kroenke* as the Search Keyword



(c) Books by Author *Kroenke* Found in the Database

FIGURE 1-1
Continued

The use of databases by Web applications and smartphone apps is illustrated in Figure 1-2. In this figure, people have computers (desktop or notebook) and smartphones, which are examples of **devices** used by people, who are referred to as **users**. On these devices are **client** applications (Web browsers, apps) used by people to obtain **services** such as searching, browsing, online purchasing, and tweeting over the Internet or cell phone networks. These services are provided by **server** computers, and these are the computers that hold the databases containing the data needed by the client applications.

This structure is known as **client-server architecture**, and it supports most of the Web applications in use today. The simple fact is that without databases, we could not have the ubiquitous Web applications and apps that are currently used by so many people.

The Characteristics of Databases

The purpose of a database is to help people keep track of things, and the most commonly used type of database is the **relational database**. We will discuss the relational database model in depth in Chapter 3, so for now we just need to understand a few basic facts about how a relational database helps people track things of interest to them.

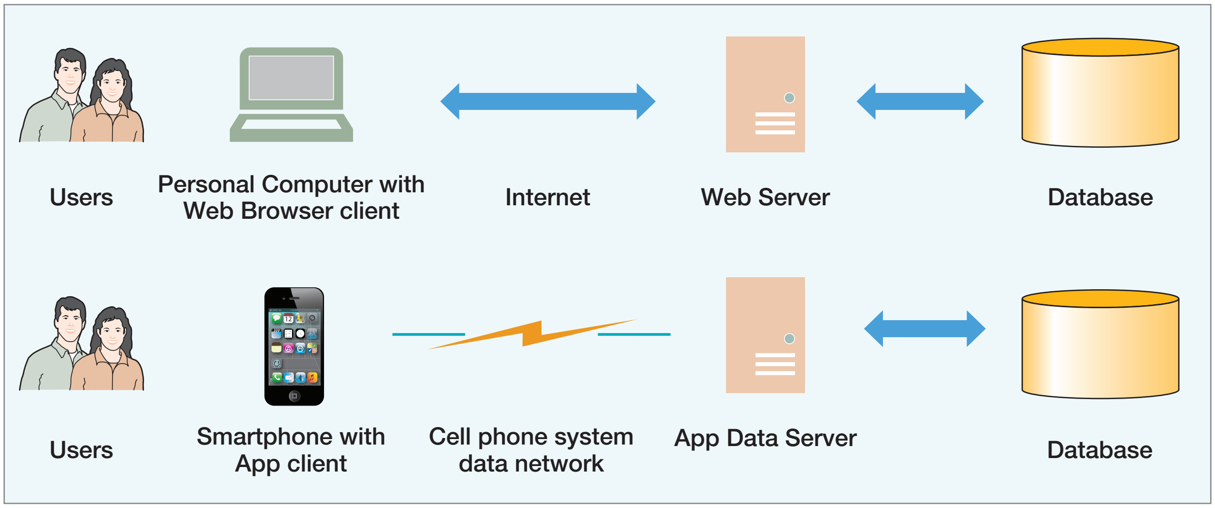
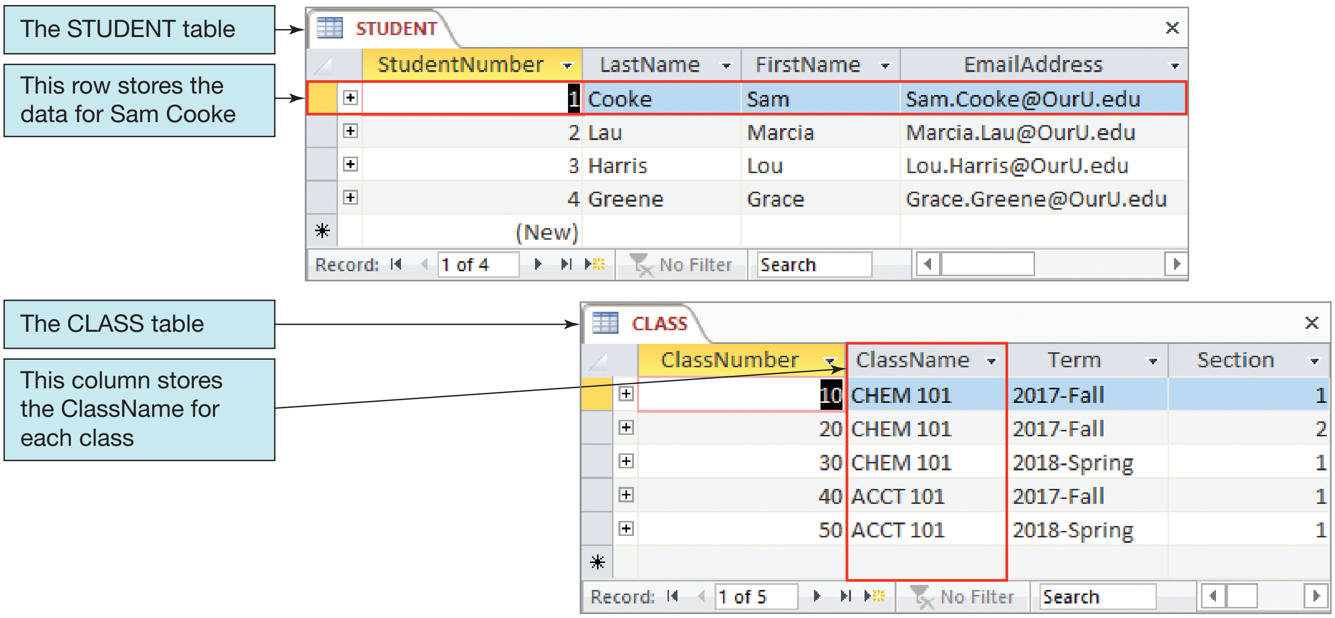


FIGURE 1-2
The Internet and Mobile Device World

A relational database stores data in tables. A **table** has rows and columns, like those in a spreadsheet. A database usually has multiple tables, and each table contains data about a different type of thing. For example, Figure 1-3 shows a database with two tables: the STUDENT table holds data about students, and the CLASS table holds data about classes.

Each **row** of a table has data about a particular occurrence, or **instance** of the thing of interest. For example, each row of the STUDENT table has data about one of four students: Cooke, Lau, Harris, and Greene. Similarly, each row of the CLASS table has data about a particular class. Because each row *records* the data for a specific instance, rows are also known as **records**. Each **column** of a table stores a characteristic common to all rows. For example, the first column of STUDENT stores StudentNumber, the second column stores LastName, and so forth. Columns are also known as **fields**.

FIGURE 1-3
The STUDENT and CLASS Tables



StudentNumber	LastName	FirstName	EmailAddress
1	Cooke	Sam	Sam.Cooke@OurU.edu
2	Lau	Marcia	Marcia.Lau@OurU.edu
3	Harris	Lou	Lou.Harris@OurU.edu
4	Greene	Grace	Grace.Greene@OurU.edu
(New)			

ClassNumber	ClassName	Term	Section
10	CHEM 101	2017-Fall	1
20	CHEM 101	2017-Fall	2
30	CHEM 101	2018-Spring	1
40	ACCT 101	2017-Fall	1
50	ACCT 101	2018-Spring	1

BY THE WAY A table and a *spreadsheet* (also known as a *worksheet*) are very similar in that you can think of both as having rows, columns, and cells. The details that define a table as something different from a spreadsheet are discussed in Chapter 3. For now, the main differences you will see are that tables have column names instead of identifying letters (for example, *Name* instead of *A*) and that the rows are not necessarily numbered.

Although, in theory, you could switch the rows and columns by putting instances in the columns and characteristics in the rows, this is never done. Every database in this text and 99.999999 percent of all databases throughout the world store instances in rows and characteristics in columns.

A Note on Naming Conventions

In this text, table names appear in capital letters. This convention will help you to distinguish table names in explanations. However, you are not required to set table names in capital letters. Microsoft Access and similar programs will allow you to write a table name as STUDENT, student, Student, or stuDent or in some other way.

Additionally, in this text column names begin with a capital letter. Again, this is just a convention. You could write the column name Term as term, teRm, or TERM or in any other way. To ease readability, we will sometimes create compound column names in which the first letter of each element of the compound word is capitalized. Thus, in Figure 1-3 the STUDENT table has columns StudentNumber, LastName, FirstName, and EmailAddress. Again, this capitalization is just a convenient convention. However, following these or other consistent conventions will make interpretation of database structures easier. For example, you will always know that STUDENT is the name of a table and that Student is the name of a column of a table.

A Database Has Data and Relationships

Figure 1-3 illustrates how database tables are structured to store data, but a database is not complete unless it also shows the relationships among the rows of data. To see why this is important, examine Figure 1-4. In this figure, the database contains all of the basic data shown in Figure 1-3 together with a GRADE table. Unfortunately, the relationships among the data are missing. In this format, the GRADE data are useless. It is like the joke about the sports commentator who announced: "Now for tonight's baseball scores: 2–3, 7–2, 1–0, and 4–5." The scores are useless without knowing the teams that earned them. Thus, a database contains both data and the relationships among the data.

Figure 1-5 shows the complete database that contains not only the data about students, classes, and grades, but also the relationships among the rows in those tables. For example, StudentNumber 1, who is Sam Cooke, earned a Grade of 3.7 in ClassNumber 10, which is Chem101. He also earned a Grade of 3.5 in ClassNumber 40, which is Acct101.

Figure 1-5 illustrates an important characteristic of database processing. Each row in a table is uniquely identified by a **primary key**, and the values of these keys are used to create the relationships between the tables. For example, in the STUDENT table StudentNumber serves as the primary key. Each value of StudentNumber is unique and identifies a particular student. Thus, StudentNumber 1 identifies Sam Cooke. Similarly, ClassNumber in the CLASS table identifies each class. If the numbers used in primary key columns such as StudentNumber and ClassNumber are automatically generated and assigned in the database itself, then the key is also called a **surrogate key**.

The STUDENT table

STUDENT

	StudentNumber	LastName	FirstName	EmailAddress
⊞	1	Cooke	Sam	Sam.Cooke@OurU.edu
⊞	2	Lau	Marcia	Marcia.Lau@OurU.edu
⊞	3	Harris	Lou	Lou.Harris@OurU.edu
⊞	4	Greene	Grace	Grace.Greene@OurU.edu
*	(New)			

Record: 1 of 4 No Filter Search

The CLASS table

CLASS

	ClassNumber	ClassName	Term	Section
⊞	10	CHEM 101	2017-Fall	1
⊞	20	CHEM 101	2017-Fall	2
⊞	30	CHEM 101	2018-Spring	1
⊞	40	ACCT 101	2017-Fall	1
⊞	50	ACCT 101	2018-Spring	1
*				

Record: 1 of 5 No Filter Search

The GRADE table—but who do these grades belong too?

GRADE

	Grade
	3.7
	3.5
	3.7
	3.1
	3.0
	3.5
*	0.0

Record: 1 of 6

FIGURE 1-4

The STUDENT, CLASS, and GRADE Tables

The STUDENT table

STUDENT

	StudentNumber	LastName	FirstName	EmailAddress
⊞	1	Cooke	Sam	Sam.Cooke@OurU.edu
⊞	2	Lau	Marcia	Marcia.Lau@OurU.edu
⊞	3	Harris	Lou	Lou.Harris@OurU.edu
⊞	4	Greene	Grace	Grace.Greene@OurU.edu
*	(New)			

Record: 1 of 4 No Filter Search

The CLASS table

CLASS

	ClassNumber	ClassName	Term	Section
⊞	10	CHEM 101	2017-Fall	1
⊞	20	CHEM 101	2017-Fall	2
⊞	30	CHEM 101	2018-Spring	1
⊞	40	ACCT 101	2017-Fall	1
⊞	50	ACCT 101	2018-Spring	1
*				

Record: 1 of 5 No Filter Search

The GRADE table with foreign keys—now each grade is linked back to the STUDENT and CLASS tables

GRADE

	StudentNumber	ClassNumber	Grade
	1	10	3.7
	1	40	3.5
	2	20	3.7
	3	30	3.1
	4	40	3.0
	4	50	3.5
*			0.0

Record: 1 of 6 No Filter Search

FIGURE 1-5

The Key Database Characteristic: Related Tables

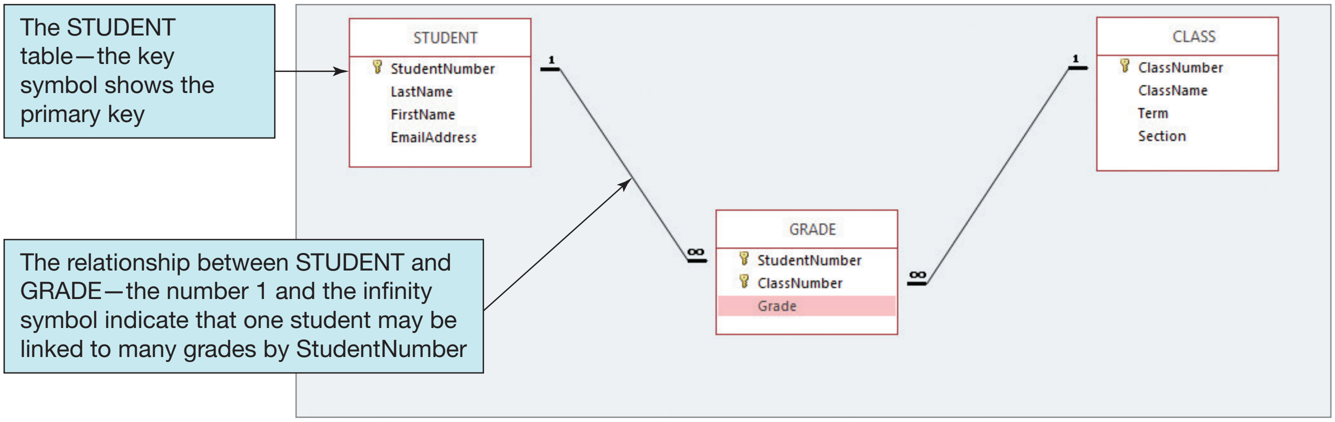


FIGURE 1-6

Microsoft Access 2016 View of Tables and Relationships

By comparing Figures 1-4, 1-5 and 1-6, we can see how the primary keys of STUDENT and CLASS were added to the GRADE table to provide GRADE with a primary key of (StudentNumber, ClassNumber) to uniquely identify each row. When more than one column in a table must be combined to form the primary key, we call this a **composite key**. More important, in GRADE StudentNumber and ClassNumber each now serves as a **foreign key**. A foreign key provides the link between two tables. By adding a foreign key, we create a **relationship** between the two tables.

Figure 1-6 shows a Microsoft Access 2016 view of the tables and relationships shown in Figure 1-5. In Figure 1-6, primary keys in each table are marked with key symbols, and connecting lines representing the relationships are drawn from the foreign keys (in GRADE) to the corresponding primary keys (in STUDENT and CLASS). The symbols on the relationship line (the number 1 and the infinity symbol) mean that, for example, one student in STUDENT can be linked to many grades in GRADE.

Databases Create Information

In order to make decisions, we need information upon which to base those decisions. Because we have already defined *data* as recorded facts and numbers, we can now define[2] **information** as:

- Knowledge derived from data
- Data presented in a meaningful context
- Data processed by summing, ordering, averaging, grouping, comparing, or other similar operations

Databases record facts and figures, so they record data. They do so, however, in a way that enables them to produce information. The data in Figure 1-5 can be manipulated to produce a student's GPA, the average GPA for a class, the average number of students in a class, and so forth. In Chapter 2, you will be introduced to a language called Structured Query Language (SQL) that you can use to produce information from database data.

To summarize, relational databases store data in tables, and they represent the relationships among the rows of those tables. They do so in a way that facilitates the production of information. We will discuss the relational database model in depth in Part 2 of this book.

[2]These definitions are from David M. Kroenke's books *Using MIS*, 8th ed. (Upper Saddle River, NJ: Prentice-Hall, 2016) and *Experiencing MIS*, 6th ed. (Upper Saddle River, NJ: Prentice-Hall, 2016). See these books for a full discussion of these definitions as well as a discussion of a fourth definition, "a difference that makes a difference."